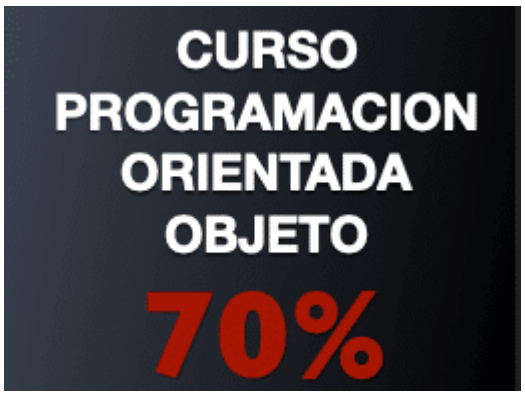
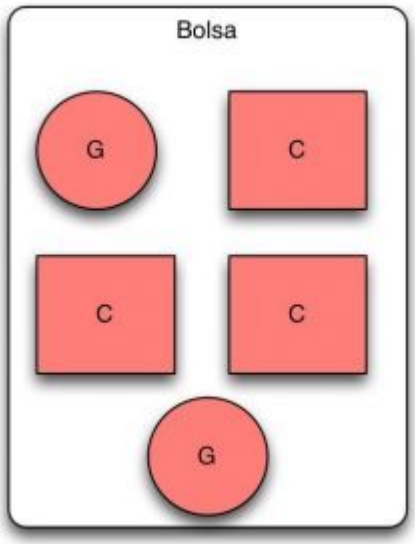
Uno de los temas que mas quebraderos de cabeza da a los desarrolladores es la construcción Java Generics o simplemente clases Genéricas.Vamos a escribir algunos artículos hablando de este tema .Para empezar con ello vamos a construir la clase Bolsa que es una clase sencilla que nos permitirá almacenar objetos de varios tipos.



Uno de los temas que mas quebraderos de cabeza da a los desarrolladores es la construcción Java Generics o simplemente clases Genéricas.Vamos a escribir algunos artículos hablando de este tema .Para empezar con ello vamos a construir la clase Bolsa que es una clase sencilla que nos permitirá almacenar objetos de varios tipos.



Esta clase tendrá un límite de objetos a almacenar . Alcanzado el limite no se podrán añadir mas .Vamos a ver su código fuente:

```
package com.arquitecturajava;
import java.util.ArrayList;
import java.util.Iterator;
public class Bolsa implements Iterable {
 private ArrayList lista = new ArrayList();
 private int tope;
 public Bolsa(int tope) {
  super();
  this.tope = tope;
 }
 public void add(Object objeto) {
  if (lista.size() >= tope) {
   lista.add(objeto);
```

```
  } else {
   throw new RuntimeException("no caben mas");
  }
 }
 public Iterator iterator() {
  return lista.iterator();
 }
}
```

En nuestro caso vamos a disponer de dos clases con las cuales rellenar la bolsa . La clase Golosina y la clase Chocolatina.

Vamos a ver su código fuente :

```
package com.arquitecturajava;
public class Chocolatina {
 private String marca;
 public String getMarca() {
  return marca;
 }
 public void setMarca(String marca) {
  this.marca = marca;
 }
 public Chocolatina(String marca) {
```

```
  super();
  this.marca = marca;
 }
}
```

```
package com.arquitecturajava;
public class Golosina {
 private String nombre;
 public String getNombre() {
  return nombre;
 }
 public void setNombre(String nombre) {
  this.nombre = nombre;
 }
 public Golosina(String nombre) {
  super();
  this.nombre = nombre;
 }
}
```

Es momento de crear un sencillo programa que llene la Bolsa de Chocolatinas y Golosinas para luego recorrer los elementos que están en la bolsa y sacarlos por pantalla.

```
package com.arquitecturajava;
public class Principal {
 public static void main(String[] args) {
  Bolsa bolsa = new Bolsa(5);
  Chocolatina c = new Chocolatina("milka");
  Chocolatina c1 = new Chocolatina("milka");
  Chocolatina c2 = new Chocolatina("ferrero");
  Golosina g1 = new Golosina("gominola");
  Golosina g2 = new Golosina("chicle");
```

```
  bolsa.add(c);
  bolsa.add(c1);
  bolsa.add(c2);
  bolsa.add(g1);
  bolsa.add(g2);
  for (Object o: bolsa) {
   if (o instanceof Chocolatina) {
    Chocolatina chocolatina = (Chocolatina) o;
    System.out.println(chocolatina.getMarca());
   } else {
    Golosina golosina = (Golosina) o;
    System.out.println(golosina.getNombre());
   }
  }
 }
}
```

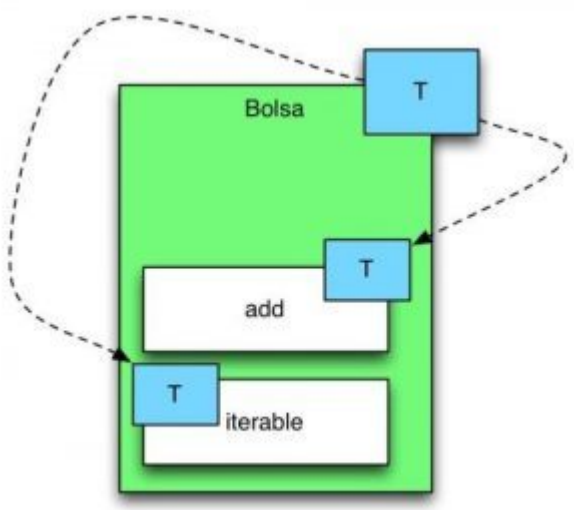
El programa funcionará correctamente, pero nos podremos dar cuenta que resulta bastante poco amigable la estructura if /else en la cual se chequean cada uno de los tipos a la hora de presentarlo por pantalla.

Java Generics

Para solventar este problema podemos construir una clase Genérica .Este tipo de clase nos permitirá definir una Bolsa de un tipo concreto . Puede ser una bolsa de Golosinas o una bolsa de Chocolatinas pero NO de las dos cosas a la vez .Esto en un principio puede parecer poco flexible pero si nos ponemos a pensar cuando programamos solemos imprimir una lista de Facturas o una lista de Compras no una lista mixta. Así pues el enfoque parece razonable. Vamos a ver el código fuente y comentarlo:

```
package com.arquitecturajava.genericos;
import java.util.ArrayList;
import java.util.Iterator;
public class Bolsa < T > implements Iterable < T > {
 private ArrayList < T > lista = new ArrayList < T > ();
 private int tope;
 public Bolsa(int tope) {
  super();
  this.tope = tope;
 }
 public void add(T objeto) {
  if (lista.size() >= tope) {
   lista.add(objeto);
  } else {
   throw new RuntimeException("no caben mas");
  }
 }
 public Iterator < T > iterator() {
  return lista.iterator();
 }
}
```

La clase es un poco peculiar ya que al no saber de entrada de que tipo va a ser la bolsa debemos declarar un tipo Genérico T a nivel de clase y que será repetido en cada uno de los métodos que lo usen.



De esta manera cuando construyamos un objeto de esta clase será el momento de especificar el tipo de Bolsa que deseamos.

En el siguiente ejemplo hemos elegido “Chocolatina” como tipo para la Bolsa. De esta manera la bolsa solo admitirá este tipo de objetos.

```
package com.arquitecturajava.genericos;

import com.arquitecturajava.Chocolatina;
import com.arquitecturajava.Golosina;
```

```
public class Principal {

 public static void main(String[] args) {

  Bolsa < Chocolatina > bolsa = new Bolsa < Chocolatina > ();
  Chocolatina c = new Chocolatina("milka");
  Chocolatina c1 = new Chocolatina("milka");
  Chocolatina c2 = new Chocolatina("ferrero");

  bolsa.add(c);
  bolsa.add(c1);
  bolsa.add(c2);

  for (Chocolatina chocolatina: bolsa) {

   System.out.println(chocolatina.getMarca());
  }

 }

}
```

Acabamos de construir nuestra primera clase Generica y hemos usado un bucle forEach para recorrerla de una forma sencilla.

Otros artículos relacionados:

- String vs StringBuffer
- Java Iterator vs ForEach
- Java Generic Erasure
- Java Generics Parte 2